

EXISTING
DATABASES AND
APPLICATIONS

FIG. 1

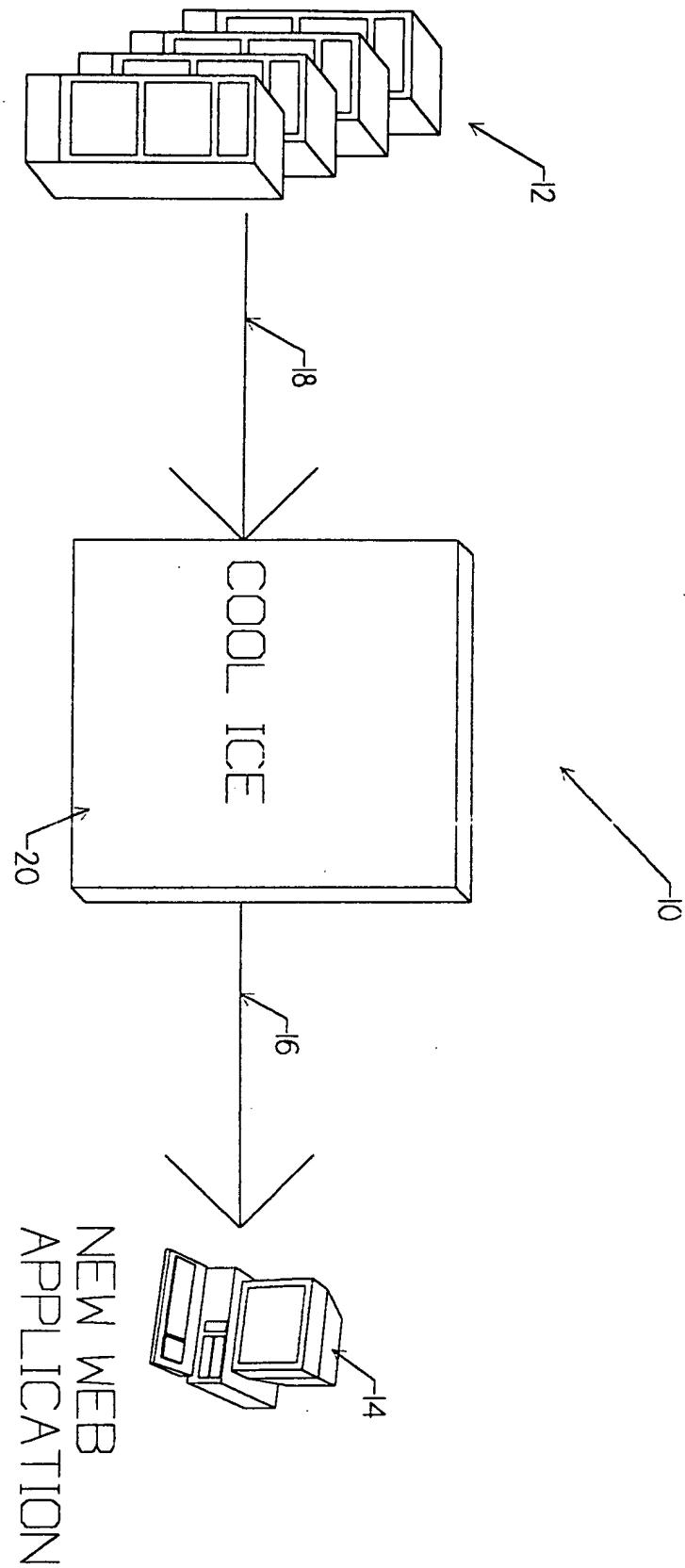


FIG. 2

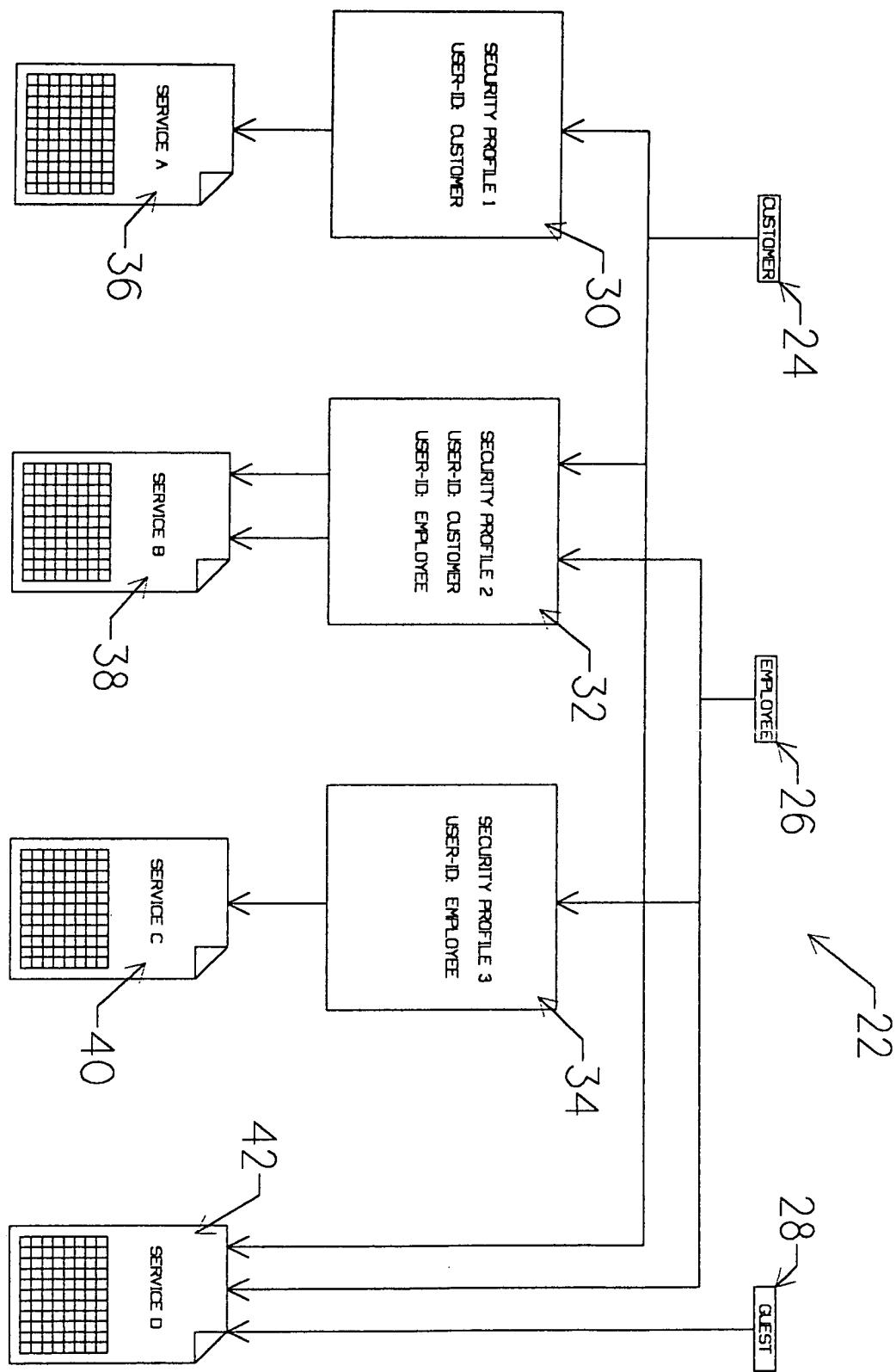


FIG. 3

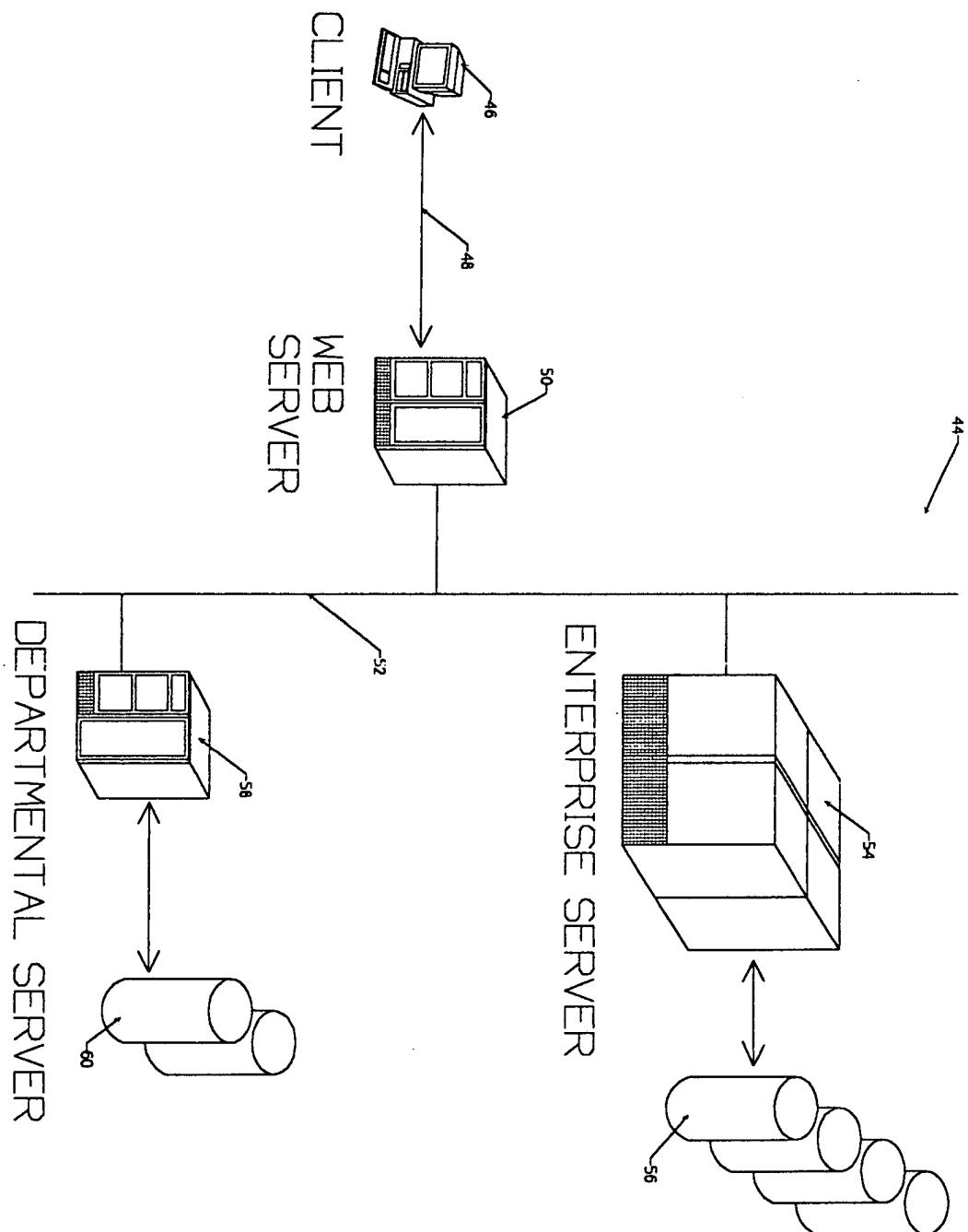
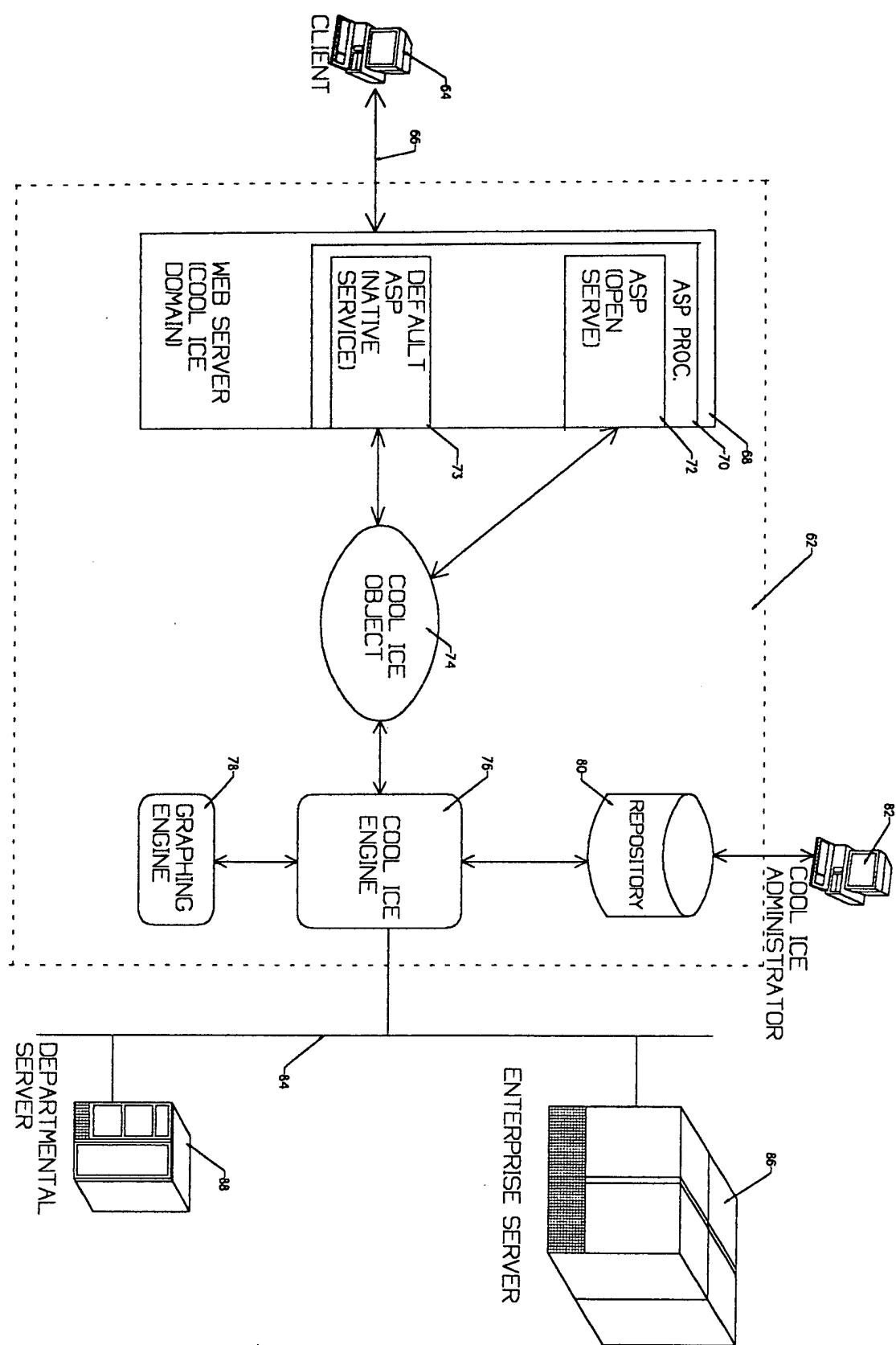
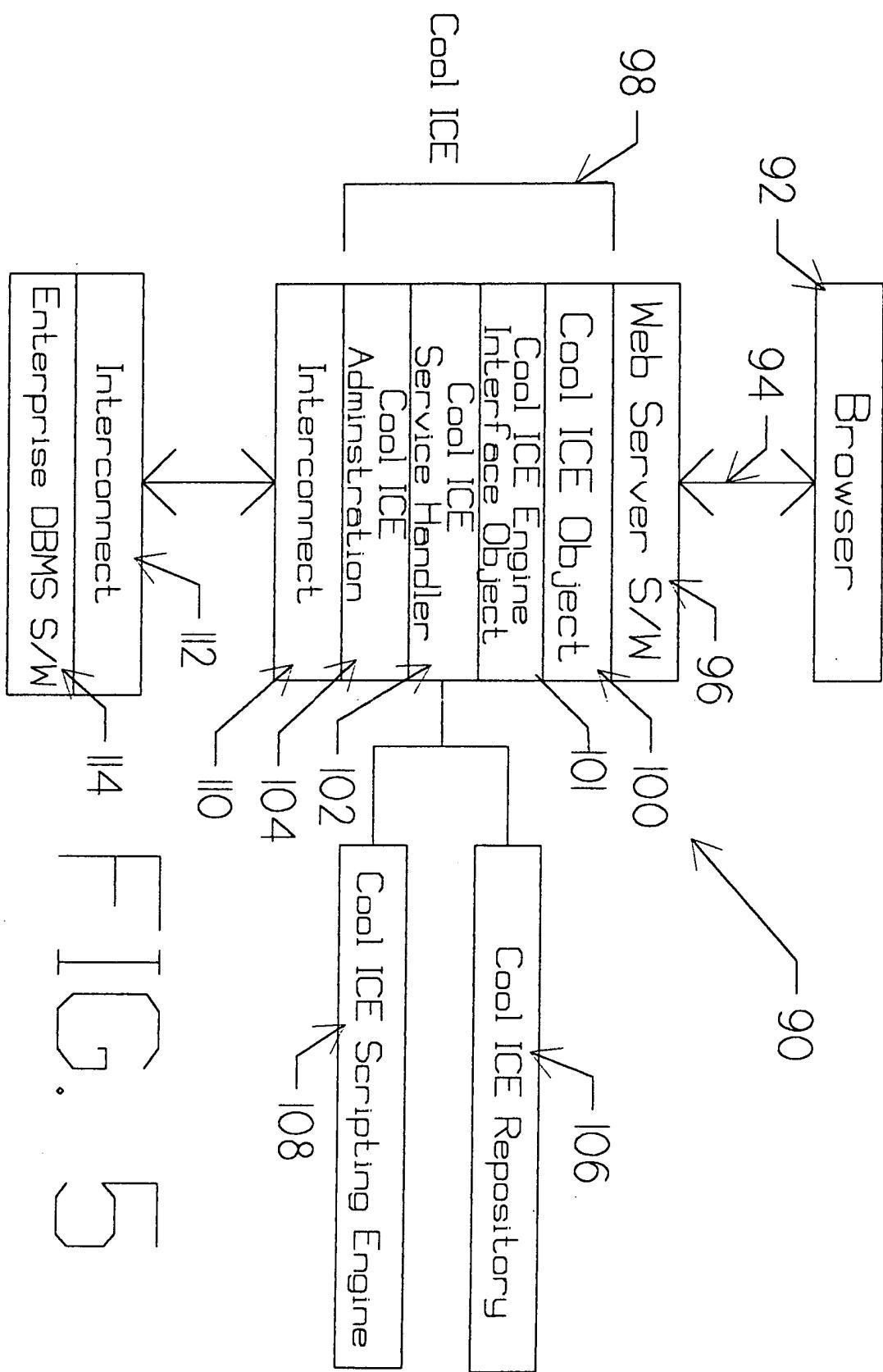


FIG. 4





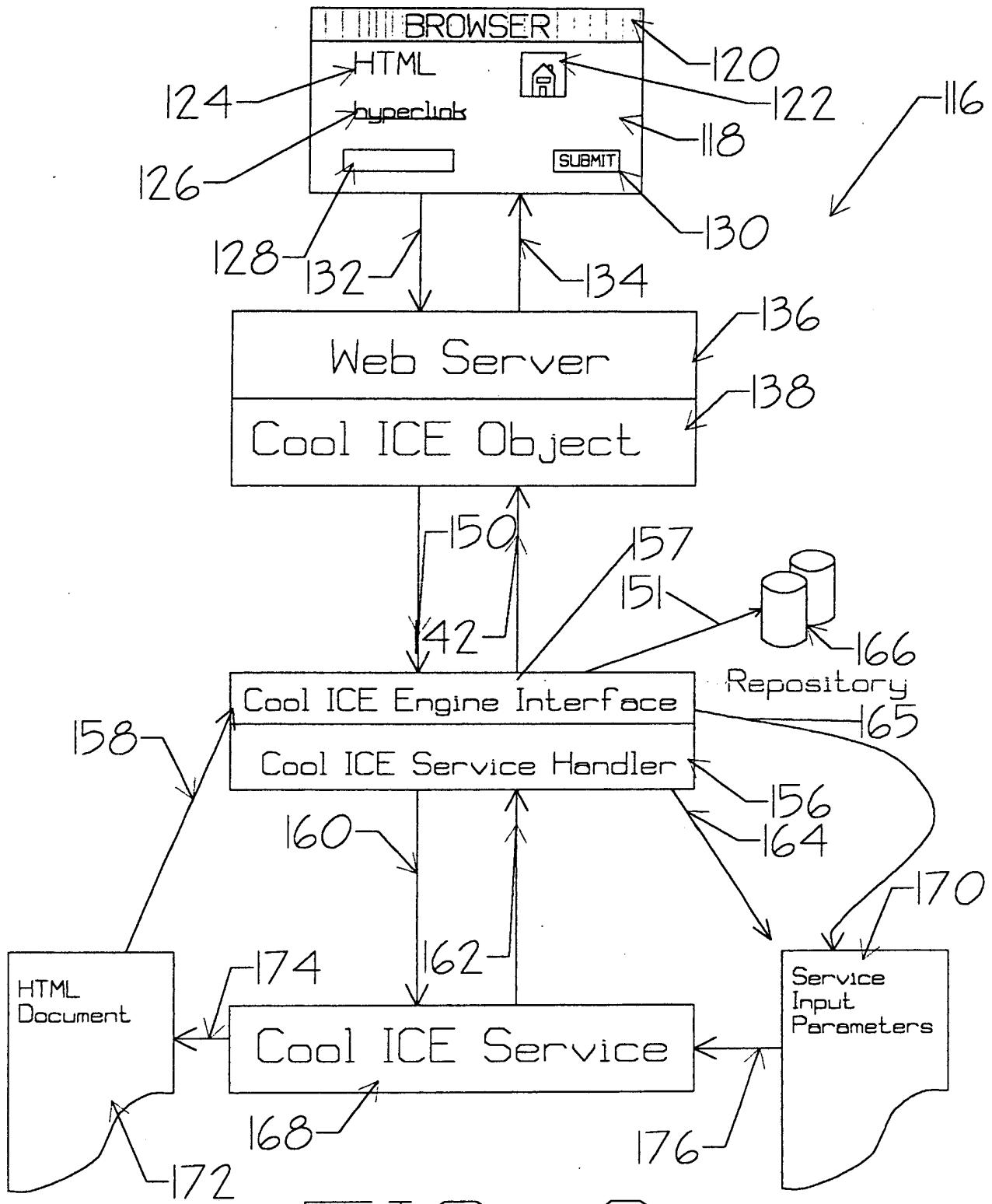


FIG. 6

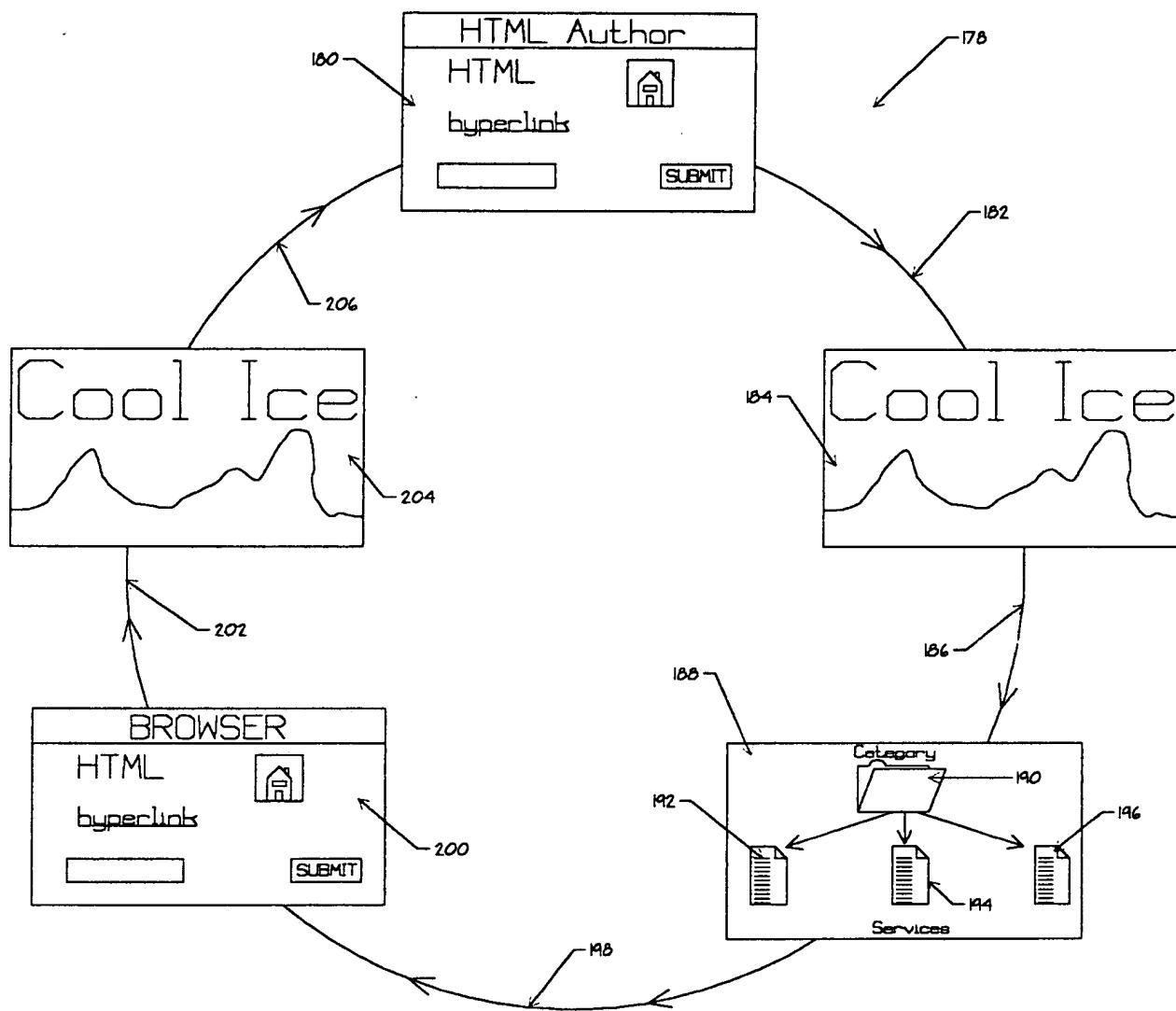
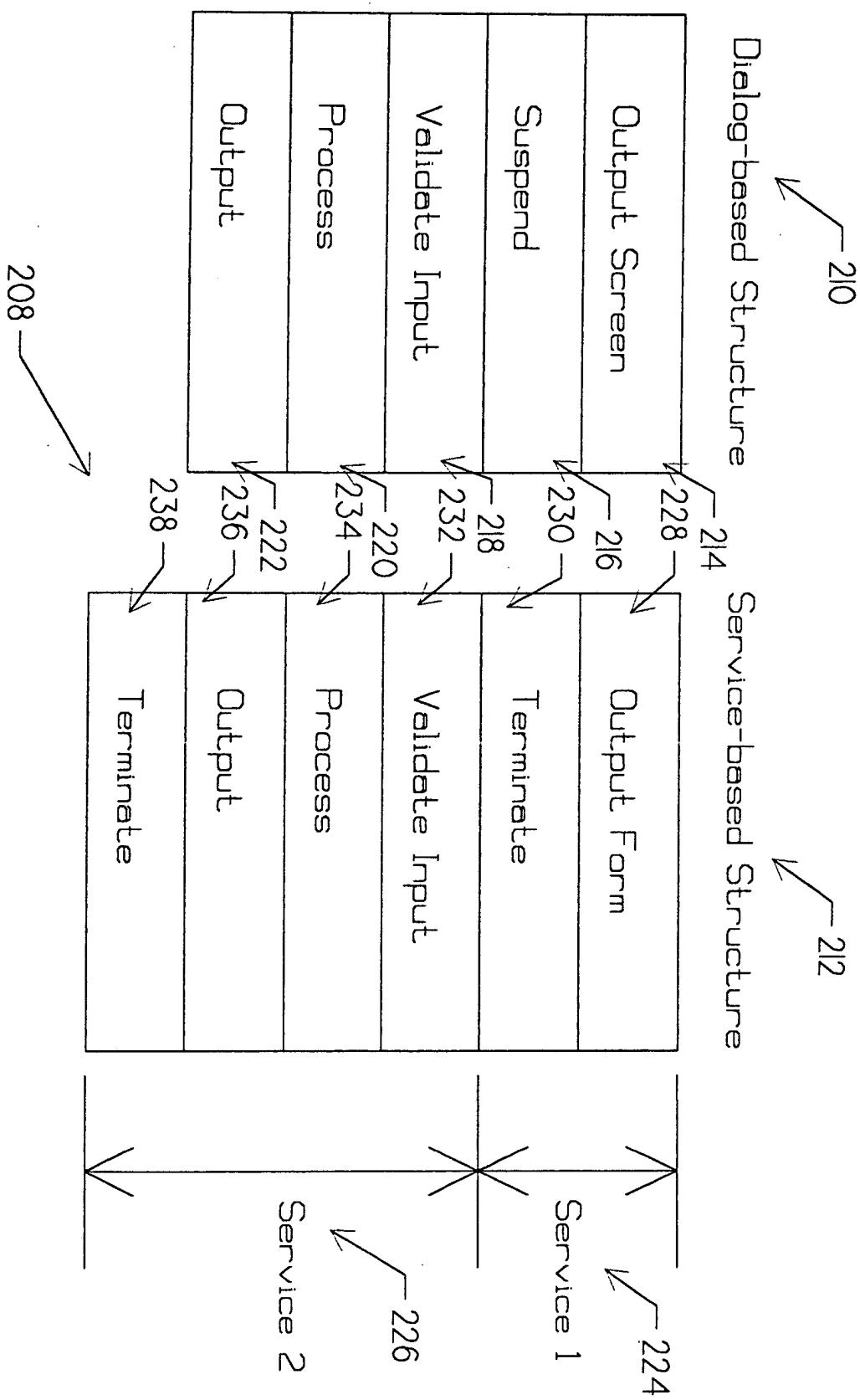
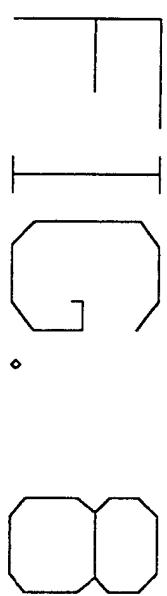


FIG. 7



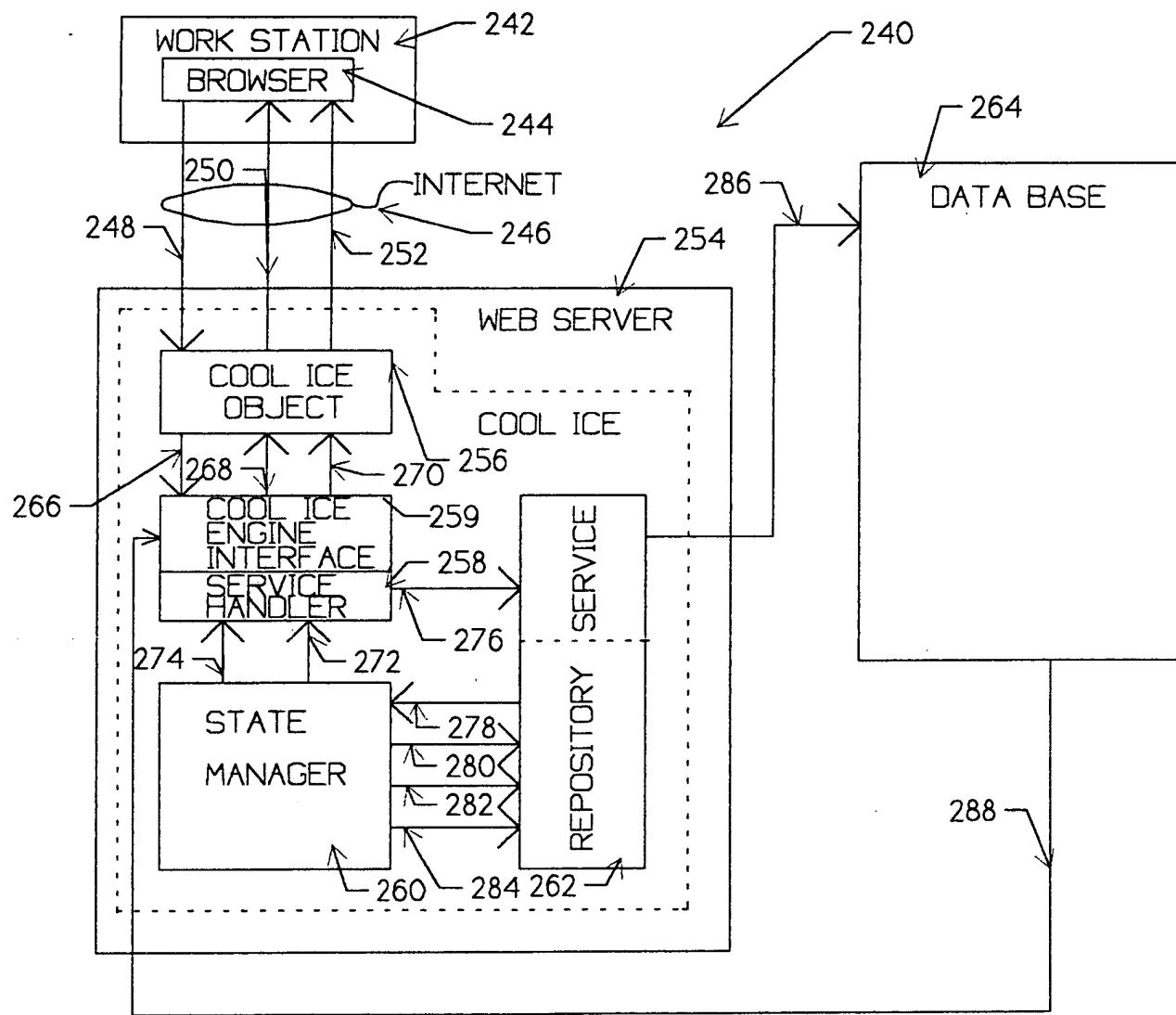


FIG. 9

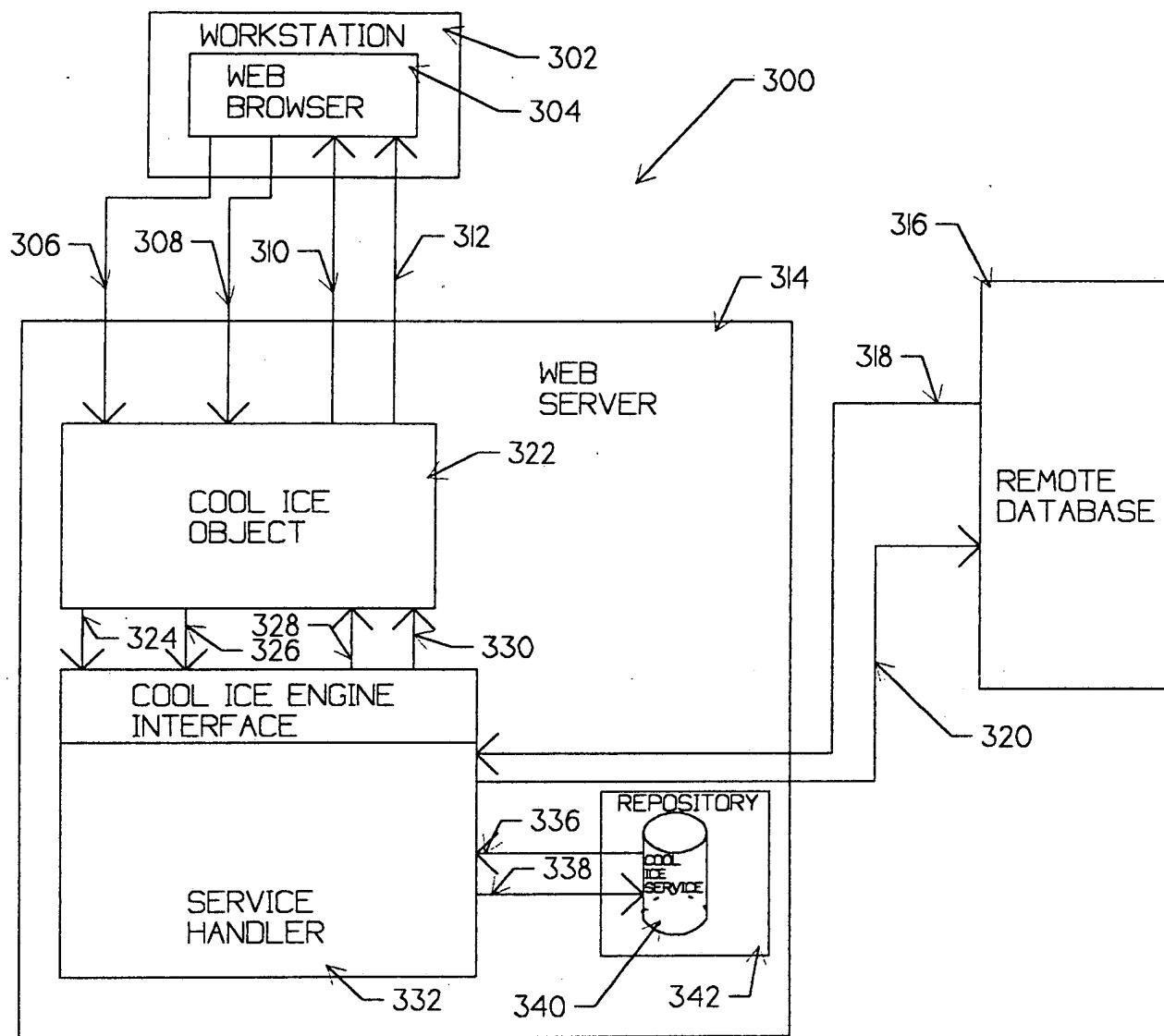


FIG. 1.

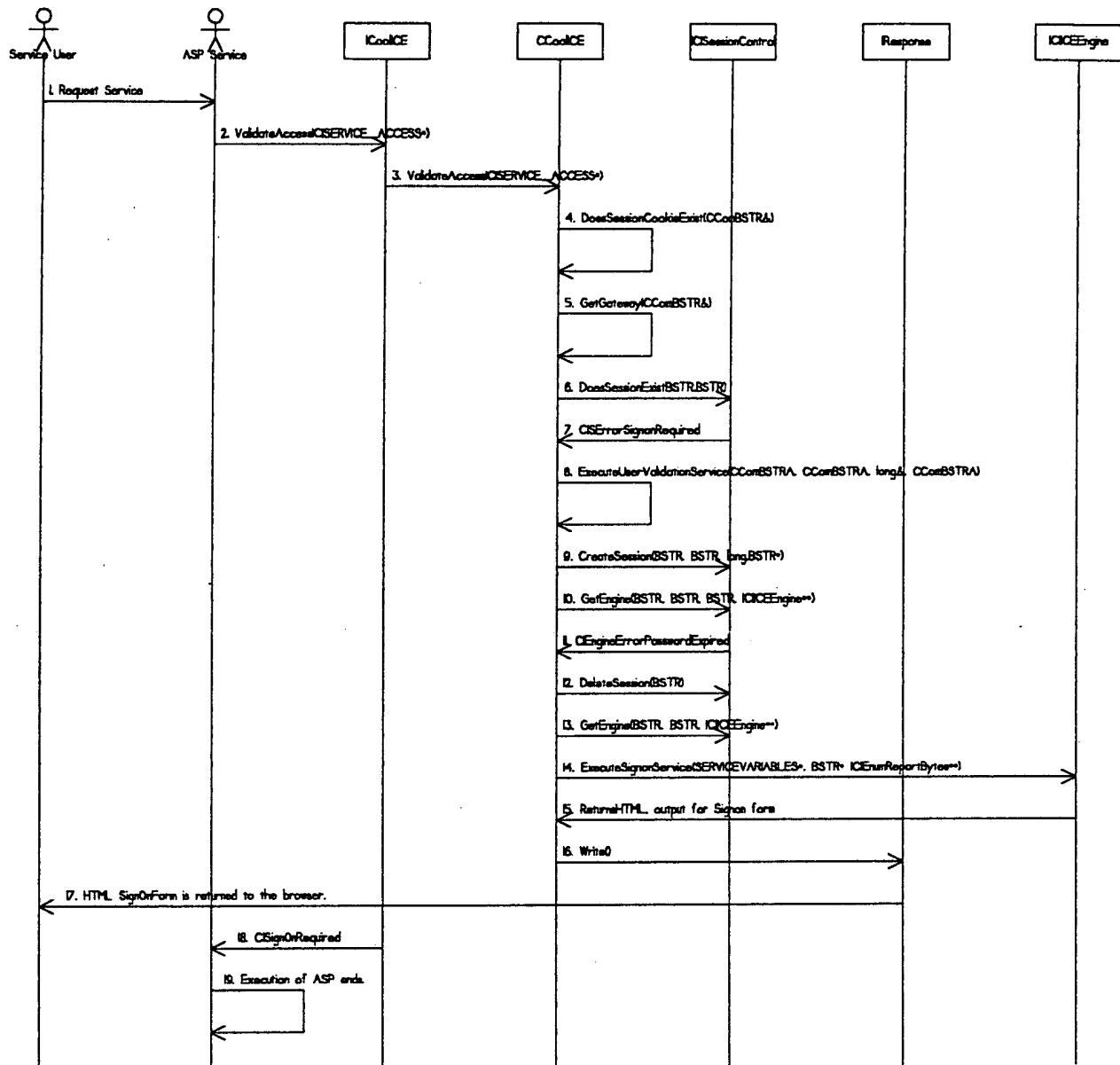


FIG. ||

BEST AVAILABLE COPY

MESSAGE #	DESCRIPTION
1.	The Service User will make a request for an ASP page from a browser.
2.	The ASP is executed which will create a CoolICE object and call the ValidateAccessMethod. This method is intended to validate that the Service User does have access to the ASP being executed.
3.	
4.	Call the helper method DoesSessionCookieExist to determine if the COSESSION cookie exists in the Request.Cookies collection. In this sequence diagram, assume S FALSE is returned.
5.	GetGateway is called to extract the Gateway Name from the ASP ServerVariables Collection of the ASP Request object. The PATH_INFO variable will return the part of the URL after the server name but before any query string. The gateway name would be the first directory in the PATH_INFO.
	For example:
	URL Request: http://MyServer/CoolICE/abc.asp PATH_INFO: /CoolICE/abc.asp Gateway: CoolICE
6.	DoesSessionExist0 is called to determine if a the HTML SignOn form needs to be processed.
	The bstrSessionID parameter is set to an empty BSTR.
	The bstrGatewayName parameter is the value returned by GetGateway0.
7.	DoesSessionExist0 has determined that a CCSession object does not exist and a sign on is required. The CCSignOnRequired HRESULT from DoesSessionExist0 is described in sequence diagrams SC02, SC04, SC05, and SC07.
8.	Call ExecuteUserValidationService to process the SignOn form input fields. In this sequence diagram assume S_DK is returned which indicates that a UserID, Department, and Password are returned. Optionally, a New Password may also be returned.
9.	CreateSession0 is called to create a CCSession object.
	The parameters are set as follows:
	-bstrGatewayName is the value returned by GetGateway0
	-bstrUserID is the value of the bstrUserID parameter from the call to ExecuteUserValidationService0
	-bstrPassword is the value of the bstrPWD parameter from the call to ExecuteUserValidationService0
	-nDept is the value of the nDept parameter from the call to ExecuteUserValidationService0
	-pstrSessionID is the address of a local variable.
10.	In order to validate that the Service User has access to the ASP, a CoolICE engine is required. In addition, if the Service User does have access, then the CoolICE engine will be needed to allow the ASP to execute additional CoolICE services.
	Therefore, CCSessionControlGetEngine0 is called to access an instance of a CoolICE engine that is managed by a Connection Pool.
	The bstrSessionID parameter is a unique identifier for the Service User. This identifier is returned by the CCSessionControlCreateSession0 method.
	The bstrGatewayName parameter is the value returned by GetGateway0.
	The bstrNewPassword parameter is the value of the bstrNewPassword parameter returned by ExecuteUserValidationService0. In most cases, this parameter will be an empty string, except when the current password has expired and a new password was specified.
11.	The call to GetEngine has returned the HRESULT value CCEnginelErrorPasswordExpired which indicates that the specified password has expired. A special SignOn form must now be displayed which allows the user to change their password.
12.	Since an Error was returned by GetEngine, it is necessary to delete the CCSession object created in step 9 above. Pass the bstrSessionID value that was returned by CreateSession0.
13.	It is necessary to get a Cool ICE Engine so that the SignOn service can be executed. In this case, a blank SessionID should be specified so that the default user-id/department/password will be used to sign on to the Cool ICE engine.
	The bstrSessionID parameter should be a blank string so that the default user-id/department/password will be used to sign on to the Cool ICE engine.
	The bstrGatewayName parameter is the value returned by GetGateway0.
	The bstrNewPassword parameter should be a blank string since we know that a new password has not been specified.
14.	The ExecuteSignOnService0 method is called to cause the SignOn service to be executed. The service will return a HTML form which has the user-id and department profiled. A field for the old password and new password must be filled in by the user.
	The input parameters 'user' and 'dept' must be passed to the SignOn service when a password has expired. The parameters are passed in the BrowserInput section of the SERVICE VARIABLES structure. The values of these parameters are the bstrUserID and nDept parameters returned by ExecuteUserValidationService0.
15.	The SignOn service returns a HTML form that prompts the user for the old password and new password. The user-id and department number fields are profiled and read-only.
16.	The HTML output for the SignOn service is passed to the Write0 method of the ASP Response object.
17.	The ASP Response object will send the HTML output for the SignOn form to the browser.
18.	ValidateAccess0 returns a CCSignOnRequired status back to the ASP. This will cause the ASP to terminate.
	Execution of the ASP is terminated due to the CCSignOnRequired status being returned from ValidateAccess0.

FIG. 12

BEST AVAILABLE COPY

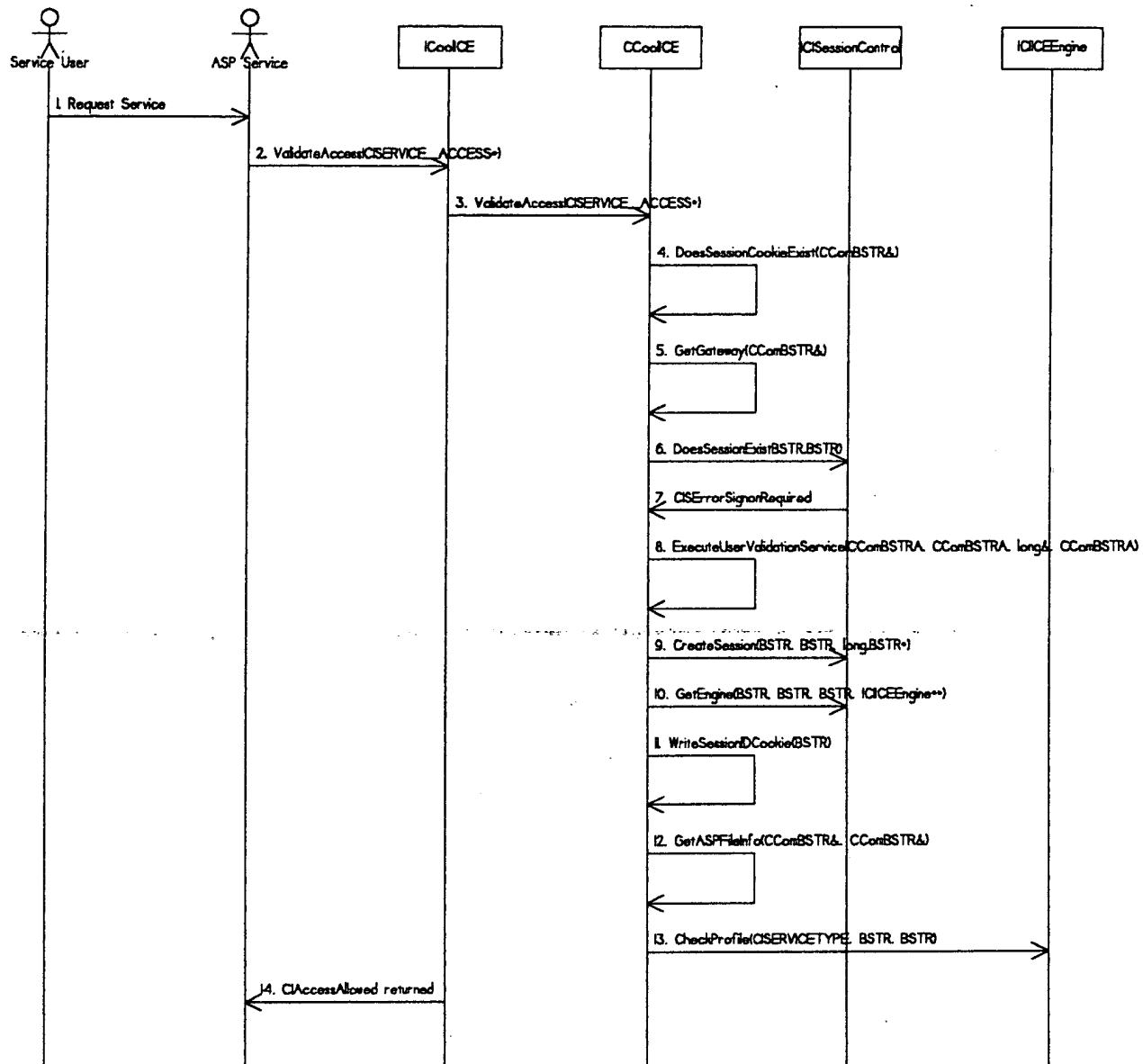


FIG. 13

BEST AVAILABLE COPY

MESSAGE	DESCRIPTION
1.	The Service User will make a request for an ASP page from a browser.
2.	The ASP is executed which will create a CoolICE object and call the ValidateAccess0 method. This method is intended to validate that the Service User does have access to the ASP being executed.
3.	
4.	Call the helper method DoesSessionCookieExist to determine if the CSESSIONID cookie exists in the Request.Cookies collection. In this sequence diagram assume S FALSE is returned.
5.	GetGateway is called to extract the Gateway Name from the ASP ServerVariables Collection of the ASP Request object. The PATH_INFO variable will return the part of the URL after the server name but before any query string. The gateway name would be the first directory in the PATH_INFO.
	For example:
	URL Request: http://MyServer/CoolICE/abc.asp PATH_INFO: CoolICE/abc.asp Gateway: CoolICE
6.	DoesSessionExists0 is called to determine if a the HTML SignOn form needs to be processed.
	The bstrSessionID parameter is set to an empty BSTR.
	The bstrGatewayName parameter is the value returned by GetGateway0.
7.	DoesSessionExists0 has determined that a CCISession object does not exist and a signon is required. The CISCErrorSignOnRequired HRESULT from DoesSessionExists0 is described in sequence diagrams SC02 SC04, SC05, and SC07.
8.	Call ExecuteUserValidationService to process the SignOn form input fields. In this sequence diagram assume S_OK is returned which indicates that a UserID, Department, and Password are returned. Optionally, a New Password is returned.
9.	CreateSession0 is called to create a CCISession object.
	The parameters are set as follows:
	-bstrGatewayName is the value returned by GetGateway0
	-bstrUserID is the value of the bstrUserID parameter from the call to ExecuteUserValidationService0
	-bstrPassword is the value of the bstrPassWd parameter from the call to ExecuteUserValidationService0
	-nDepartment is the value of the nDept parameter from the call to ExecuteUserValidationService0
	-pbstrSessionID is the address of a local variable.
10.	In order to validate that the Service User has access to the ASP, a CoolICE engine is required. In addition, if the Service User does have access, then the CoolICE engine will be needed to allow the ASP to execute additional CoolICE services.
	Therefore, ICISessionControlGetEngine0 is called to access an instance of a CoolICE engine that is managed by a Connection Pool.
	The bstrSessionID parameter is a unique identifier for the Service User. This identifier is returned by the ICISessionControlCreateSession0 method.
	The bstrGatewayName parameter is the value returned by GetGateway0.
	The bstrNewPassword parameter is the value of the bstrNewPassword parameter returned by ExecuteUserValidationService0. In most cases, this parameter will be an empty string, except when the current password has expired and a new password was specified.
11.	Call WriteSessionIDCookie to write the SessionID value returned by ICISessionControlCreateSession out to the browser as the CSESSIONID cookie.
12.	The helper method GetASPFileInfo is called to retrieve the virtual directory alias name and the file name of the ASP.
13.	The ICIEEngineCheckProfile0 method is called to verify that the user, as known to the Cool ICE engine, does have access to the ASP.
14.	The ValidateAccess0 method will return CIAccessAllowed status indicating that the Service User does have access, therefore, the execution of the ASP can continue.

FIG. 14

BEST AVAILABLE COPY